

PLP - Recuperatorio del Segundo Parcial - 1^{er} cuatrimestre de 2022

Este examen se aprueba obteniendo al menos dos ejercicios bien menos (B-) y uno regular (R). Las notas para cada ejercicio son: -, I, R, B-, B. Entregar cada ejercicio en hojas separadas. Poner nombre, apellido, número de orden y cantidad de hojas en la primera hoja, y numerar las hojas. Se puede utilizar todo lo definido en las prácticas y todo lo que se dio en clase, colocando referencias claras. El orden de los ejercicios es arbitrario. Recomendamos leer el parcial completo antes de empezar a resolver.

Ejercicio 1 - Programación Lógica

Implementar los predicados respetando en cada caso la instanciación pedida. Los generadores deben cubrir todas las instancias válidas de aquello que generan sin repetir dos veces la misma. No usar cut (!) ni predicados de alto orden como `setof`, con la única excepción de `not`.

- a) Definir el predicado `montaña(+L, -L1, -C, -L2)` que es verdadero cuando la lista `L` es una montaña, es decir, sus elementos crecen hasta una cima, y luego decrecen (estrictamente). En `L1`, `C` y `L2` deben instanciarse la sublista creciente, el elemento de la cima, y la sublista decreciente respectivamente (las listas no deben ser vacías). Por ejemplo:

```
?- montaña([1,2,3,1], L1, C, L2).
```

```
L1 = [1, 2], C = 3, L2 = [1] ;
```

```
false.
```

```
?- montaña([1,2,2], L1, C, L2).
```

```
false.
```

- b) Analizar la reversibilidad de los parámetros `L1`, `C` y `L2` (por separado, no es necesario hacer todas las combinaciones de instanciaciones posibles).
- c) Definir el predicado `todasLasListas(+A, -L)` que es verdadero cuando `L` es una lista de elementos del alfabeto `A`. Notar que en `L` deben instanciarse **todas** las posibles listas (puede haber elementos repetidos). Por ejemplo:

```
?- todasLasListas([a,b,c], L).
```

```
L = [a] ;
```

```
L = [b] ;
```

```
L = [c] ;
```

```
L = [a,a] ;
```

```
L = [b,a] ;
```

```
L = [c,a] ;
```

```
L = [a,b] ;
```

```
...
```

Aclaración: la implementación no necesariamente debe mostrar los resultados en el mismo orden que en el ejemplo. Notar que el predicado debe devolver infinitas soluciones. Pueden asumir que `A` no es vacía y no tiene elementos repetidos.

Ejercicio 2 - Resolución

Considerar las siguientes definiciones en prolog:

```
permutacion([], []).
```

```
permutacion([X|XS], P) :- permutacion(XS, YS), sacar(X, P, YS).
```

```
sacar(X, [X|XS], XS).
```

```
sacar(X, [Y|XS], [Y|YS]) :- sacar(X, XS, YS).
```

y la consulta:

```
?- permutacion([1,2], [2,1]).
```

a) Convertir la base de conocimientos y la consulta a forma clausal.

Sugerencia: pueden pensar la lista [1,2] como [1|[2|[]]]

b) Utilizar resolución SLD para probar que la consulta es verdadera.

Ejercicio 3 - Cálculo de Objetos

Sean los siguientes objetos:

$$\begin{aligned} \text{true} &\stackrel{\text{def}}{=} [\text{not} = \varsigma(t)[\text{not} = t, \text{if} = \lambda(x)\lambda(y)y], \text{if} = \lambda(x)\lambda(y)x] \\ \text{false} &\stackrel{\text{def}}{=} [\text{not} = \text{true}, \text{if} = \lambda(x)\lambda(y)y] \\ \text{cero} &\stackrel{\text{def}}{=} [\text{iszero} = \text{true}, \text{pred} = \varsigma(x)x, \text{succ} = \varsigma(x)(x.\text{iszero} := \text{false}).\text{pred} := x] \\ \text{uno} &\stackrel{\text{def}}{=} [\text{iszero} = \text{false}, \text{pred} = \text{cero}, \text{succ} = \varsigma(x)(x.\text{iszero} := \text{false}).\text{pred} := x] \\ \text{o} &\stackrel{\text{def}}{=} [\text{val} = \varsigma(f)\lambda(n)(n.\text{iszero}).\text{if}(f.\text{arg})((f.\text{val}(n.\text{pred})).\text{succ}), \text{arg} = \varsigma(f)f.\text{arg}] \end{aligned}$$

(Recordar la codificación del Cálculo Lambda vista en clase: $[[\lambda x.M]] \stackrel{\text{def}}{=} [\text{val} = \varsigma(y)[[M]]\{y.\text{arg}/x\}, \text{arg} = \varsigma(y)y.\text{arg}]$)

a) ¿Qué representa el objeto o? Proponer un nombre mejor y explicar con palabras cómo funciona.

b) Mostrar cómo reduce la siguiente expresión: o (uno) (cero)

Se recomienda definir objetos auxiliares. Se permite utilizar la regla APP que aparece en la guía.